BREU MANUAL DE LOGO

Maria Guilera Almirall
Santiago Manrique Catalán

1 PRIMITIVES

1.1 GRÀFIQUES

1.1.1 ELEMENTALS

inicia.dibuix
id

Serveix per a esborrar els dibuixos de la pantalla abans de fer-ne de nous i per a tornar a situar la Tortuga al mig i orientada cap amunt.

avança
av

Aquesta ordre ha d'anar seguida sempre d'un número. Fa avançar la Tortuga el nombre d'unitats indicades, en la direcció que assenyali el seu cap. S'ha de deixar, almenys, un espai en blanc entre l'ordre i el número.

Exemples:

```
?avança 50
?inicia.dibuix  avança 100
?inicia.dibuix  avança 20 avança 30 avança 50
```

gira.dreta
gd

Fa girar el cap de la Tortuga vers la dreta el nombre d'unitats que s'indiqui tot seguit. Cal escriure el nombre de graus a girar després de l'ordre, deixant almenys un espai en blanc entremig.

Exemples:

```
?gira.dreta 90
?inicia.dibuix gira.dreta 270
?inicia.dibuix avança 50 gira.dreta 90 avança 30
?inicia.dibuix gira.dreta 360
?inicia.dibuix gira.dreta 45 avança 100 gira.dreta  135
avança 70 gira.dreta 90 avança 200
```

gira.esquerra
ge

Fa girar el cap de la Tortuga vers l'esquerra el nombre d'unitats que s'indiqui tot seguit. Cal escriure el nombre de graus a girar després de l'ordre, deixant almenys un espai en blanc entremig.

Exemples:

```
?gira.esquerra 90
?inicia.dibuix gira.esquerra 200
?inicia.dibuix avança  40  gira.esquerra  90  avança 40
gira.dreta 90 avança 50
```

recula
re

Fa recular la Tortuga el nombre d'unitats que s'indiqui, en la direcció que assenyali el seu cap (la Tortuga va enrera). Aquesta ordre ha d'anar seguida del nombre d'unitats. S'ha de deixar, almenys, un espai en blanc entre l'ordre i el número.

Exemples:

```
?recula 40
```

?inicia.dibuix recula 50 gira.dreta 90 recula 50
?inicia.dibuix gira.dreta 45 avança 100 recula 100 gira.esquerra 90 avança 100 recula 100 gira.dreta 45

esborra.dibuix Esborra la pantalla gràfica mantenint a la Tortuga en la posició i direcció que tenia abans de donar l'ordre. A vegades és útil per aconseguir aspectes d'animació.

1.1.2 COLOR

En el moment en què s'entra en el LOGO gràfic el color del fons de la pantalla és negre i el del llapis cian; no hi haurà cap canvi fins que l'usuari no doni les ordres necessàries. Si el color del llapis és el mateix que el del fons no es pot veure el dibuix. Els colors s'identifiquen per mitjà d'un codi de color numèric. L'ús del color a la pantalla de text s'explica més endavant.

fes.color
fc Ha d'anar seguit d'un número que representi el codi del color que es vol assignar al llapis. Un mateix número, entre el 0 i el 3, donarà un color o un altre depenent de quin sigui l'estat de la paleta. Al principi el color és el 3. Els colors, segons les paletes, són:

Codi	Paleta 0	Paleta 1
0	fons	fons
1	verd	cian
2	vermell	magenta
3	marró	blanc

fes.paleta Ha d'anar seguit d'un número. Fixa la paleta en una de les dues opcions 0 o 1. Al principi, la paleta és la 1. Quan es modifica la paleta canvia automàticament el color del llapis.

fes.fons
ff Ha d'anar seguit d'un número (del 0 al 15) que representa el color del fons gràfic. El codis corresponents són:

0: negre	4: vermell	8: gris	12: vermell clar
1: blau	5: magenta	9: blau clar	13: magenta clar
2: verd	6: marró	10: verd clar	14: groc
3: cian	7: blanc	11: cian clar	15: blanc lluminós

color Retorna el número que representa el codi del color actual del llapis.

paleta Retorna el número que representa la paleta actual.

fons Retorna el número que representa el color actual del fons gràfic.

omple Omple una forma tancada que conté en el seu interior la Tortuga amb el color actual del llapis. Si la Tortuga es troba sobre una línia vertical o horitzontal, la tornarà a dibuixar en el color actual del llapis. Si es troba sobre una línia obliqua no farà res. Per poder omplir una zona, la Tortuga ha d'estar en modalitat **llapis**.

Exemple:

```
procediment omplir.quadrat
repeteix 4 [avança 100 gira.dreta 90]
no.llapis gira.dreta 45 avança 15  llapis
omple
no.llapis recula 15 gira.esquerra 45 llapis
fi
```

1.1.3 MODALITATS DEL LLAPIS

llapis Baixa el llapis de la Tortuga de manera que quan aquesta es mou deixa un rastre sobre la pantalla o sigui que dibuixa. **llapis** no admet cap número a continuació. Quan s'engega el LOGO gràfic, la Tortuga es troba en modalitat **llapis**. Aquest estat es manté fins que es dóna una de les ordres alternatives: **no.llapis, goma** o **llapis.invers.**

no.llapis Aixeca el llapis de la Tortuga de manera que en moure's no deixa cap rastre o dibuix. **no.llapis** no requereix ni admet cap argument. Aquesta modalitat es manté fins que es dóna una ordre alternativa: **llapis, goma** o **llapis.invers**

Exemple:

```
?avança 100 gira.dreta 90 no.llapis avança 50
gira.dreta 90 llapis avança 100
```

goma Permet esborrar tot el que s'ha dibuixat amb el llapis. En passar, en aquesta modalitat, sobre una zona dibuixada s'esborrarà. Es desactiva amb **llapis, no.llapis** o **llapis.invers.**

llapis.invers Transforma el llapis de la Tortuga en un 'inversor' de color. Quan la Tortuga es desplaça inverteix els colors per allà on passa (intercanvia els colors del fons i els del llapis), per tant deixa rastre allà on no hi era i l'esborra allà on estava dibuixat. Es desactiva amb **llapis, no.llapis** o **goma.**

1.1.4 ESTATS DE LA TORTUGA

desapareix
dap

Serveix per amagar la Tortuga. Això permet que la velocitat d'execució de les ordres augmenti, encara que es perd informació en no veure's nia posició ni l'orientació de la Tortuga. **desapareix** no requereix ni admet cap argument. Aquest estat es manté fins que es dóna l'ordre contrària **apareix.**

apareix
ap

Es l'ordre contrària a l'anterior i fa que es visualitzi la Tortuga. És l'estat inicial en què es troba el sistema. No necessita cap argument.

Exemples:
?inicia.dibuix desapareix avança 50 gira.dreta 90 avança 50
?apareix
?inicia.dibuix desapareix gira.dreta 90
?apareix.

1.1.5 GEOMETRIA DE COORDENADES

1.1.5.1 COORDENADES CARTESIANES

El LOGO també permet treballar amb coordenades cartesianes. Cada punt de la pantalla queda identificat per les seves coordenades. La primera coordenada representa la coordenada horitzontal i la segona la vertical. Si es treballa amb pantalla mixta la coordenada x varia entre -160 i 160. La coordenada y entre -50 i 111. Sobre pantalla gràfica les unitats són: coordenada x entre -160 i 160, coordenada y entre -111 i 111. El centre físic de la pantalla té les coordenades (0,0). Aquestes unitats poden variar, ja que depenen dels valors de les proporcions escollides.

1.1.5.2 ORDRES AMB COORDENADES CARTESIANES

posa't [x y]

Mou la Tortuga fins a la posició (x,y) deixant rastre si està en estat **llapis.** No modifica la seva orientació. L'argument ha de ser una llista numèrica.

Exemples:
?posa't [-23 41] mou la Tortuga fins a la posició (-23,41)
?posa't (frase :x 23) la mou fins a la posició (:x,23)

fes.x n

Mou la Tortuga horitzontalment fins que la coordenada x prengui el valor donat n. Si l'estat és **llapis** dibuixarà una línia. No modifica la seva orientació.

Exemples:
?fes.x 30 mou la Tortuga horitzontalment fins a x=30
?fes.x :costat la mou horitzontalment fins a x=:costat

fes.y n Mou la Tortuga verticalment fins que la coordenada y prengui el valor donat n. Si l'estat és llapis dibuixarà una línia. No modifica la seva orientació.

Exemples:
?fes.y -5 mou la Tortuga verticalment fins a y=-5
?fes.y :altura la mou verticalment fins a y=:altura

punt [x y] Dibuixa el punt de coordenades (x,y), sense moure la Tortuga, en la posició indicada per la llista. Requereix un únic argument que ha de ser una llista de dos números. El punt és del color actual del llapis.

Exemples:
?punt [-3 -8] dibuixa un punt en la posició (3,-8)
?punt (frase :x :y+7) el dibuixa en la posició (:x,:y+7)

1.1.5.3 FUNCIONS SOBRE COORDENADES CARTESIANES

posició Retorna la posició actual de la Tortuga expressada com a una llista de dos números: la coordenada horitzontal i la coordenada vertical.

Exemple:
posició retorna [0 0] si la Tortuga es troba a l'origen de coordenades

coor.x Retorna la coordenada x que correspon a la posició actual de la Tortuga.

Exemple:
coor.x retorna 11 si la Tortuga és a la posició (11,-21)

coor.y Retorna la coordenada y que correspon a la posició actual de la Tortuga.

Exemple:
coor.y retorna -21 si la Tortuga és a la posició (11,-21)

color.punt [x y] Retorna el codi que correspon al color del punt indicat a la llista.

Exemple:

color.punt [50 15] pot retornar 0, 1, 2 o 3.

1.1.5.4 ORIENTACIÓ DE LA TORTUGA

orienta't n — Orienta el cap de la Tortuga en la direcció indicada per n (angle expressat en graus). Els angles es comencen a comptar des de l'orientació nord i en sentit horari.

Exemples:
orienta't 0 posa la Tortuga mirant cap amunt
orienta't 90 posa la Tortuga mirant cap a la dreta

orientació — Retorna l'orientació de la Tortuga. Un número entre 0 i 360.

Exemple:
orientació retorna 180 si la Tortuga mira cap a baix

vers [x y] — Retorna l'orientació del punt (x,y) vist des de la posició de la Tortuga.

Exemple:
vers [-5 0] retorna 270 si la Tortuga era al centre de la pantalla (0,0).

1.1.6 ALTRES PRIMITIVES GRÀFIQUES

a.casa — No requereix cap argument. Porta la Tortuga al centre de la pantalla i la deixa mirant cap amunt. Si el seu estat és llapis deixa rastre per allà on passa.

Exemple:

```
procediment triangle :costat1 :angle :costat2
a.casa
avança :costat1 gira.dreta :angle avança :costat2
a.casa
fi
```

fes.llapis llista — Fixa l'estat del llapis d'acord amb les dades d'una llista de tres elements que representen respectivament: modalitat del llapis, codi del seu color i codi de la paleta.

Exemple:
fes.llapis [llapis.invers 2 0]

estat.llapis Retorna una llista de tres elements que descriuen les característiques actuals del llapis de la Tortuga. El primer indica la modalitat del llapis, el segon indica el codi del color i el tercer indica el codi de la paleta.

Exemple:
escriu estat.llapis
goma 3 1

fes.forma n Dóna a la Tortuga la forma corresponent a un dels codis ASCII compresos entre el 0 i el 127 o a una forma amb codi posterior (entre 128 i 255) donada a través de la primitiva **guarda.forma**. Una vegada canviada la forma no es poden veure les variacions d'orientació de la Tortuga. Perquè torni a tenir la forma original cal indicar **fes.forma** 256.

forma No requereix cap argument. Retorna el número associat a la forma actual de la Tortuga.

guarda.forma n Pren com a forma de la Tortuga la figura sobre la qual es troba aquesta i li assigna un codi n (codi comprès entre el 128 i el 255). Es pot aconseguir que la Tortuga prengui aquesta forma utilitzant la primitiva **fes.forma**.

estampa No requereix cap argument. Deixa una imatge fixa de la Tortuga, segons la seva forma i color actual, allà on es troba.

1.2 PANTALLES

1.2.1 TIPUS DE PANTALLA

Quan es treballa amb LOGO la pantalla pot estar en tres estats diferents: sols text, gràfics més text ó sols gràfics. Les pantalles també es poden classificar com a pantalla d'alta resolució i pantalla de baixa resolució. Tot seguit es descriu breument cadascun dels estats i s'indica com accedir-hi.

text Tota la pantalla està dedicada a text. És l'estat inicial del sistema i l'estat al qual es passa en sortir de l'editor. També s'hi pot accedir pitjant la tecla de funció **F1**.

mixt Tota la pantalla està dedicada a dibuix menys les set últimes línies que es reserven per a text. També s'hi pot accedir pitjant la tecla de funció **F2**.

gràfic Tota la pantalla està dedicada a gràfics i no es poden veure les ordres, però sí els missatges d'error. També s'hi pot accedir pitjant la tecla de funció F4.

Quan es treballa en les pantalles mixta o gràfica existeixen dues alternatives bàsiques. Es pot escriure a 80 columnes o bé a 40. En el primer cas, els textos s'escriuen a 80 columnes, els gràfics són d'alta resolució i es disposa tan sols de dos colors per als dibuixos. En el segon cas, els textos s'escriuen a 40 columnes, els gràfics són de resolució mitjana i es disposa de quatre colors. L'elecció d'una de les dues modalitats es fa mitjançant la primitiva fes.columnes.

fes.columnes Ha d'anar seguit d'un número N, comprès entre el 2 i el 80 (Generalment 40 o 80). Fixa l'amplada de la pantalla en N columnes.

1.2.2 PANTALLA GRÀFICA

El pla per on es mou la Tortuga es pot ajustar de tres maneres diferents amb les ordres:

tor És l'estat en què es troba inicialment la Tortuga. L'extrem superior de la pantalla toca i continua sobre l'extrem inferior; el lateral dret toca i continua sobre l'esquerre. Es com si la Tortuga es mogués sobre un pneumàtic.

finestra La Tortuga es mou lliurement dins un pla il.limitat del qual tan sols veiem una part (com si es mirés per una finestra). La Tortuga pot desaparèixer de la pantalla fàcilment.

barrera La Tortuga es mou dintre de la zona del pla que coincideix amb la pantalla. Si s'intenta de fer-la fora de la pantalla apareix el missatge: la tortuga se'n va fora!.

El LOGO té dues ordres que serveixen per ajustar la relació entre les unitats horitzontals i les verticals de la pantalla, evitant possibles deformacions de les figures:

fes.proporció Ha d'anar seguida d'una llista [x y]. El primer element serveix per fixar les unitats horitzontals i el segon per a les verticals. Ambdós números han d'estar compresos entre 1 i 200.

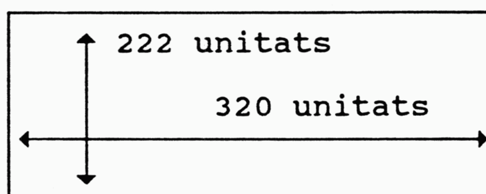
Exemple:

?fes.proporció [100 45]

proporció Retorna l'escala que està fixada actualment, en forma de llista de dos números. El primer representa l'escala horitzontal i el segon la vertical.

Les unitats de la pantalla gràfica depenen del valor de **proporció**. En la pantalla de 40 columnes la **proporció** per defecte és [100 90] i es divideix en 320 x 222 unitats. En la pantalla de 80 columnes la **proporció** per defecte és [200 90] i les unitats són les mateixes:

pantalla gràfica



1.2.3 <u>PANTALLA DE TEXT</u>

La pantalla de text té 25 files. El nombre de columnes pot ser 40 o 80, encara que no es pot escriure a les columnes 39 o 79, segons el cas.

pantalla de 40 columnes

[0 0]	[39 0]
[0 24]	[39 24]

pantalla de 80 columnes

[0 0]	[79 0]
[0 24]	[79 24]

esborra.text
esbt Esborra la pantalla de text. Torna a col.locar el cursor a l'extrem superior esquerre de la pantalla.

mou.cursor Requereix com a argument una llista de dos elements. El primer és el que correspon al número de la columna i el segon el que correspon al de la fila. Col.locarà el cursor en la posició indicada.

<u>Exemples</u>:
?mou.cursor [10 21]
?mou.cursor frase :columna :fila

cursor No necessita cap argument. Retorna la posició en forma d'una llista de dos elements que indiquen, respectivament, la columna i la fila sobre les quals es troba el cursor.

fes.color.text
fct Ha d'anar seguit d'un número que representi un codi de color. Canvia el color del text que s'escriu a partir del moment en què s'ha donat l'ordre. (Vegeu codis de color a la pàgina 4).

fes.fons.text
fft Ha d'anar seguit d'un número que representi un codi de color. Canvia el color de fons dels caràcters que s'escriuen a partir del moment en què s'ha donat l'ordre.

color.text Retorna el número que indica el color del text.

fons.text Retorna el número que indica el color del fons del text

1.3 NÚMEROS, PARAULES I LLISTES

1.3.1 PRIMITIVES D'EXTRACCIÓ

primer Requereix un argument. Si s'aplica a un paraula retorna el seu primer caràcter. Si s'aplica a una llista retorna el seu primer element.

Exemples:
primer "solució retorna "s
primer [exemple de llista] retorna "exemple
primer [[1 un][2 dos]] retorna [1 un]

últim Requereix un argument. Si s'aplica a una paraula retorna el seu últim caràcter. Si s'aplica a una llista retorna el seu últim element.

Exemples:
últim "solució retorna "ó
últim [exemple de llista] retorna "llista
últim [[1 un] [2 dos]] retorna [2 dos]

sense.primer
sp Requereix un argument. Aplicat a una paraula retorna aquesta paraula però sense el seu primer caràcter. Aplicat a una llista retorna aquesta llista però sense el seu primer element.

Exemples:
sense.primer "solució retorna "olució
sense.primer [exemple de llista] retorna [de llista]
sense.primer [[1 un] 5 [2 dos]] retorna [5 [2 dos]]

sense.últim
su

Requereix un argument. Aplicat a una paraula retorna aquesta paraula però sense el seu últim caràcter. Aplicat a una llista retorna aquesta llista però sense el seu últim element.

Exemples:
sense.últim "solució retorna "soluci
sense.últim [exemple de llista] retorna [exemple de]
sense.últim [[1 un] 5 [2 dos]] retorna [[1 un] 5]

element n objecte
el

Retorna l'element que ocupa el lloc n dins de l'objecte (objecte pot ser una paraula o una llista). n no pot superar el número de elements de l'objecte.

Exemples:
element 3 [exemple de llista] retorna "llista
element 1 [[a e] [i o] u] retorna [a e]
element 5 "paraula retorna "u

1.3.2 PRIMITIVES DE COMPOSICIÓ

paraula

Usualment requereix dos arguments, que han de ser dues paraules, que retorna unides formant una paraula única. Si el nombre d'arguments és diferent de dos, tota l'expressió ha d'anar entre parèntesis.

Exemples:
paraula "solu "ció retorna "solució
paraula 19 92 retorna 1992
(paraula "so "lu "ció) retorna "solució

frase

Usualment requereix dos arguments que poden ser: dues llistes, una paraula i una llista o bé dues paraules (les paraules són considerades com a llistes d'un sol element). Els retorna units formant una llista única. Si el nombre d'arguments és diferent de dos, tota l'expressió ha d'anar entre parèntesis.

Exemples:
frase [això és][una llista] retorna [això és una llista]
frase [un cas] "mixt retorna [un cas mixt]
(frase [a e] "i [o u]) retorna [a e i o u]
frase [[0] 1 2] [3 4] retorna [[0] 1 2 3 4]

llista

Retorna una llista formada pels arguments que se li han donat. Es diferencia de frase en què no fusiona totes les llistes en una, sinó que respecta l'estructura prèvia dels arguments.

Exemples:

llista [això és] [una llista] retorna [[això és][una llista]]
(llista [a e] "i [o u]) retorna [[a e] i [o u]]
llista [[0] 1 2] [3 4] retorna [[[0]1 2] [3 4]]

anteposant
apo

Requereix dos arguments, el primer pot ser una paraula o una llista i el segon ha de ser, necessàriament, una llista. Retorna una nova llista en la qual el primer element serà el primer argument.

Exemples:
anteposant "a [m n p] retorna [a m n p]
anteposant [a b] [m n p] retorna [[a b] m n p]
anteposant [1 un] [[2 dos][3 tres]] retorna [[1 un] [2 dos] [3 tres]]

posposant
ppo

Semblant a l'anterior però retorna una nova llista en la qual l'últim element serà el primer argument.

Exemples:
posposant "x [m n p] retorna [m n p x]
posposant [x y] [m n p] retorna [m n p [x y]]

1.3.3 ALTRES PRIMITIVES

n.elements objecte
nel

Retorna un número. Si objecte és una paraula retorna el número de caràcters que la formen. Si objecte és una llista retorna el número d'elements que la composen.

Exemples:
n.elements [exemple de llista] retorna 3
n.elements [[a e] i [o u]] retorna 3
n.elements "paraula retorna 7

ascii objecte

L'objecte ha de ser una paraula. Retorna el codi ascii del primer caràcter de la paraula. Aquesta pot quedar reduïda a un sol caràcter però no pot ser la paraula buida. Consultar el manual per a tenir els números del codis.

Exemples:
ascii "a retorna 65
ascii "barba retorna 66

caràcter n

Requereix un argument que ha de ser un nombre enter comprès entre 0 i 255. Retorna una paraula formada per el caràcter corresponent al codi ASCII del número donat.

Exemple:

carácter 65 retorna "a

1.3.4 OPERADORS LÒGICS

veres.totes
vt

Operador de conjunció. Retorna **ver** si tots els seus arguments són certs, en cas contrari retorna **fals.** Requereix, per defecte, dos arguments; si aquest nombre no és dos, tota l'expressió ha d'anar entre parèntesis.

Exemple:
veres.totes (:costat>0) (:costat<100)

vera.alguna
va

Operador de disjunció. Retorna **fals** si tots els seus arguments són falsos a la vegada, en cas contrari retorna **ver.** Requereix, per defecte, dos arguments; si aquest nombre no és dos, tota l'expressió ha d'anar entre parèntesis.

Exemple:
(vera.alguna (:ter= "ar) (:ter="er) (:ter="ir))

no

Operador de negació. Retorna el valor lògic contrari. Únicament requereix un argument.

Exemple:
no (:costat=0)

1.3.5 FUNCIONS LÒGIQUES

pertany

Requereix dos arguments. Retorna **ver** si el primer argument (caràcter, paraula o llista) és un element del segon argument. En cas contrari retorna **fals.** Cal recordar que un caràcter és un element d'una paraula, una paraula és un element d'una llista i una llista és un element d'una llista de llistes.

Exemples:
pertany "a "mama retorna ver
pertany [a b] [a b c] retorna fals
pertany [a b] [[a b] c] retorna ver

és.nombre

Requereix un argument. Retorna **ver** si l'argument donat és un nombre. En cas contrari retorna **fals.**

Exemple:
és.nombre 34 retorna ver

és.buida Requereix un argument. Retorna **ver** si l'argument donat és la paraula buida o la llista buida. En cas contrari retorna **fals**.

Exemples:
és.buida [] retorna ver

tecleig Retorna **ver** si s'ha polsat alguna tecla. En cas contrari retorna **fals**.

També es disposa, entre d'altres, de les primitives: **és.llista** i **és.paraula**. (Vegeu manual)

1.3.6 FUNCIONS MATEMÀTIQUES

Totes les funcions matemàtiques que es donen tot seguit requereixen arguments numèrics.

arctg x Retorna l'arctangent de x (angle en graus).

Exemple:
arctg 1 retorna 45

arrel x Retorna l'arrel quadrada de x.

Exemple:
arrel 9 retorna 3

arrodonit x Retorna el valor de x arrodonit a la unitat més propera.

Exemples:
arrodonit 4.25 retorna 4
arrodonit 4.83 retorna 5

atzar x Genera un número aleatori comprés entre 0 i x-1.

Exemples:
atzar 6 pot retornar 0, 1, 2, 3, 4 o 5.
1 + atzar 3 pot retornar 1, 2 o 3

cos x Retorna el cosinus de l'angle x (x en graus).

Exemple:
cos 90 retorna 0

part.entera x Retorna la part entera de x.

Exemples:
part.entera 4.25 retorna 4 retorna

part.entera 2 retorna 2
part.entera -5.78 retorna -6

potència x y Pren x com a base, y com a exponent i retorna x^y. Tan x com y poden ser nombres naturals, enters o reals.

Exemple:
potència 2 3 retorna 8

quocient x y Retorna el quocient de la divisió entera entre x i y.

Exemple:
quocient 23 5 retorna 4

residu x y Retorna el residu de la divisió entera entre x i y.

Exemple:
residu 23 5 retorna 3

sin x Retorna el sinus de l'angle x (x en graus).

Exemple:
sin 90 retorna 1

També es disposa, entre d'altres, de les primitives: **exp** (exponencial de base e), **ln** (logaritme neperià) i **pi** (3,14159...).(Vegeu manual)

1.4 ENTRADA I SORTIDA D'INFORMACIÓ

1.4.1 SORTIDA D'INFORMACIÓ

escriu
esc
Requereix un argument. Visualitza sobre la pantalla el número, la paraula o la llista que s'indiquen a continuació d'aquesta ordre. Si el nombre d'arguments és diferent d'un, cal posar-ho tot entre parèntesis; els diferents elements s'escriuran separats per espais en blanc. En acabar l'execució el cursor es col.loca al començament de la línia següent.

Exemples:
?escriu "gener escriurà gener
?(escriu [avui és] "dilluns) escriurà avui és dilluns

escriu.seguit
ess
Anàleg a l'anterior però escriu els elements sense deixar espais en blanc entre ells. Manté el cursor en la posició següent a la de l'últim caràcter escrit.

Exemples:
?(escriu.seguit [avui és] "dilluns) escriurà avui ésdilluns

mostra — Actua de manera anàloga a **escriu** amb l'única diferència que les llistes s'escriuen entre claudàtors.

Exemples:
?mostra "paraula escriurà paraula
?mostra [una llista] escriurà [una llista]

1.4.2 ENTRADA D'INFORMACIÓ

car.llegit
cl — Produeix una interrupció del procediment. Retorna, en forma de paraula, el caràcter corresponent a la tecla que s'ha polsat. Després de polsar-la no s'ha de pitjar ←┘. No es visualitza per pantalla el caràcter entrat, que passa directament a la memòria de treball.

Exemple:
```
procediment contestador
escriu [Escriu el teva primera inicial]
posa.a "p car.llegit
escriu [Escriu la teva segona inicial]
posa.a "q car.llegit
escriu [Escriu la teva tercera inicial]
posa.a "r car.llegit
(escriu :r :p :q  [no són les teves inicials])
fi
```

paraula.llegida
pl — Produeix una interrupció en el procediment. Retorna el text teclejat en forma de paraula. Cal teclejar ←┘ per indicar la fi de la paraula. Per pantalla apareix el que es va introduint. No s'han de posar cometes al començament de la paraula.

Exemple:
```
procediment edat
escriu [Indica la teva edat per mitjà d'un número]
posa.a "edat paraula.llegida
escriu [L'any passat tenies :edat - 1 anys]
fi
```

llista.llegida
ll — Produeix una interrupció del procediment. Retorna el text teclejat en forma de llista. Per a indicar que s'ha acabat l'entrada cal polsar ←┘. Per pantalla es pot veure el que es va entrant. No s'ha de posar claudàtors en teclejar la llista.

Exemple:
```
procediment salutació
```

```
escriu [tecleja el teu nom i cognoms]
posa.a "resposta llista.llegida
(escriu [bon dia] primer :resposta)
fi
```

1.5 ASSIGNACIÓ I CONTROL

1.5.1 ASSIGNACIÓ

posa.a "var objecte Permet donar valors a les variables. En aquest cas objecte s'assignarà a la variable var. Es útil tant per a donar valors inicials, com per a modificar-los al llarg d'un procés (per exemple dins d'una ordre **repeteix**). Es pot utilitzar per a guardar valors numèrics, paraules o llistes.

Notació per a les variables. Si el nom de la variable és **var**

:var representa el valor que conté la variable

"var designa el nom de la variable

Exemples:

?posa.a "n 5 assigna a la variable n el valor 5
?posa.a "n :n-1 assigna a la variable n el valor que tingués menys una unitat
?posa.a "p sense.primer "hola assigna a la variable p la paraula ola

fes.local Limita l'àmbit d'una variable al procediment on es troba, evitant interferències amb variables del mateix nom existents en altres procediments. Es recomana el seu ús quan la variable s'ha creat amb un **posa.a**.

Exemple:

```
procediment quadrats
fes.local "costat
posa.a "costat 10
repeteix 3 [quadrat :costat posa.a "costat :costat+10]
fi
```

1.5.2 REPETICIÓ

repeteix n [accions] Permet de repetir una acció o conjunt d'accions un nombre determinat de vegades. L'acció o accions que s'han de repetir han d'anar dins de claudàtors. El nombre de repeticions s'ha de col.locar entre el **repeteix** i el primer claudàtor, amb una separació mínima d'un espai en blanc.

Exemples:
```
?repeteix 4 [avança 50 gira.dreta 90]
?repeteix 99 [escriu [arbre s'escriu sense h]]
```

1.5.3 CONDICIONALS

si si expressió.lògica [acció1] [acció2]
si expressió.lògica [acció]

Si l'expressió lògica dóna el valor **ver** (és certa) s'executa l'acció 1 i en cas contrari l'acció 2. Després es continua amb l'execució del procediment. Cal teclejar tota la línia seguida i polsar ↵ després de tancar els segons claudàtors. Es pot omitir [acció2].

Exemples:
```
?si :número < 0 [acaba]
?si :llista = [] [escriu "despedida acaba]
?si :meu > :teu [escriu [és més gran]] [escriu [és més
petit]]
```

comprova expressió.lògica Guarda el resultat de l'expressió lògica (ver o fals)

si.ver [acció1] Si el resultat de l'últim **comprova** és **ver**
sv s'executen les accions descrites dins dels claudàtors.

si.fals [acció2] Si el resultat de l'últim **comprova** és **fals**
sf s'executen les accions descrites dins dels claudàtors.

Exemples:
```
procediment acaba.en.a :paraula
comprova últim :paraula = "a
si.ver [escriu [acaba en a]]
si.fals [escriu [no acaba en a]]
fi
```

1.5.4 INTERRUPCIÓ DE PROCEDIMENTS

acaba Detura el procediment en curs i retorna el control al procediment que l'ha cridat o a la forma interactiva. Només té sentit utilizar-lo dins d'un procediment.

retorna objecte Aquesta instrucció, dins un procediment, finalitza l'execució d'aquest i envia el valor calculat al procediment que l'ha cridat. Requereix un argument que pot ser un número, paraula o llista. Les instruccions que es troben darrera d'un retorna no s'executen. Permet la creació de funcions.

<u>Exemples</u>:
```
procediment hipotenusa :x :y
retorna arrel :x * :x + :y * :y
fi
```

1.5.5 <u>DETENCIÓ DE PROCEDIMENTS</u>

espera n Requereix un argument numèric. Atura una execució durant n pulsacions del rellotge, la qual cosa permet llegir resultats i observar certs efectes que d'altra manera passarien desapercebuts. (Aproximadament, en el Micral-30, un segon equival a 18 pulsacions)

<u>Exemples</u>:
```
?repeteix 12 [avança 50 espera 40 goma recula 50 llapis
gira.dreta 30]
```

```
procediment lletrejar
escriu [Les vocals són]
posa.a "vocals [a e i o u]
repeteix 5  [escriu primer "vocals posa.a "vocals
sense.primer :vocals espera 50]
fi
```

continua Reprèn l'execució d'un procediment després de pausa o després de polsar **F5**. No es pot utilitzar dins d'un procediment.

pausa S'utilitza únicament dins d'un procediment. Interromp la seva execució al mateix temps que envia un missatge. A partir d'aquest moment es retorna a la forma interactiva i novament es poden donar ordres. continua serveix per a reemprendre l'execució del procediment a partir del punt en que es va deturar.

1.6 <u>ALTRES PRIMITIVES</u>

1.6.1 <u>MANIPULACIÓ DE PROPIETATS</u>

Les primitives que es descriuen a continuació serveixen per a la manipulació de les llistes de propietats.

assigna.propietat nom prop val   Assigna a nom la propietat prop amb un valor val (nom i prop han de ser paraules).

<u>Exemples</u>:
?assigna.propietat "poma "color "vermell
fa correspondre a poma la característica color amb el valor vermell.
?assigna.propietat "poma "tipus "camosa

?assigna.propietat "poma "origen "Lleida
Després d'aquesta tercera assignació la poma és de color vermell, de tipus camosa i d'origen Lleida.
?assigna.propietat "plàtan "origen [Illes Canàries]

treu.propietat nom prop Esborra la propietat associada al nom i també, per tant, el seu valor corresponent.

Exemple:
?treu.propietat "poma "color
Deixa poma tan sols com de tipus camosa i d'origen Lleida.

oblida.propietats nom Esborra totes les propietats associades a nom. Cal tenir en compte que **oblida.tot** no esborra les propietats.

Exemple:
?oblida.propietats "poma
Deixa poma sense cap propietat.

propietat nom prop Retorna el valor de la propietat associada al nom.

Exemples:
propietat "poma "tipus retorna "camosa
propietat "poma "color retorna []

propietats nom Retorna la llista de totes les propietats associades al nom.

Exemple:
propietats "poma retorna [tipus camosa origen Lleida]

propietat? nom.de.grup o llista Retorna les propietats del grup o de la llista.

Exemples:
propietat? [poma plàtan]
assigna.propietat "poma "color "vermell
assigna.propietat "poma "origen "Lleida
assigna.propietat "plàtan "origen [Illes Canàries]

1.6.2 AGRUPACIÓ EN PAQUETS

Les opcions que es descriuen tot seguit són interessants ja que permeten la creació de subconjunts o grups de procediments dins de la memòria de treball i la seva organització posterior d'acord amb les necessitats de la programació. La possibilitat

'd'amagar' grups de procediments permet obtenir recursos didàctics: el professor pot definir procediments que de cara l'alumne tenen el mateix tractament que les primitives del LOGO.

agrupa "nom.del.grup [llista de procediments]

Assigna tots els elements de la llista al paquet que té per nom 'nom.del.grup'. Tota acció referida al nom del grup, s'extendrà a tots i cada un dels procediments que el composen.

Exemple:
?agrupa "figures [triangle rectangle quadrat pentàgon]

agrupa.tot "nom.del.grup Assigna al nom del grup tots els procediments i variables que estan dins l'espai de treball en un moment donat (i que no pertanyen a un altre grup).

protegeix "nom.del.grup Amaga tots els procediments i les variables que pertanyen al grup. Tan sols actua sobre grups.

Exemple:
?protegeix "figures
a partir d'aquest moment no es poden veure les definicions dels procediments amagats.

desprotegeix "nom.del.grup La seva acció és contrària a la de l'ordre anterior. Permet visualitzar novament els procediments i variables prèviament amagats. Actua sobre grups.

Exemple:
?desprotegeix "figures

1.6.3 PRIMITIVES D'ÚS AVANÇAT

executa llista Necessita un argument que és una llista i que s'executa com si les ordres que conté fossin entrades per teclat.

Exemple:
?posa.a "quad [repeteix 4 [avança 50 gira.dreta 90]]
?executa :quad
dibuixarà un quadrat

contingut nom Retorna el valor corresponent al seu argument. El signe : de les variables és una abreviatura de contingut; contingut "quadrat equival a :quadrat.

<u>Exemples</u>:
```
?posa.a "dia3  "dilluns
?escriu contingut "dia3
dilluns
?escriu contingut paraula "dia 3
dilluns
```

defineix nom llista Permet definir procediments. Requereix dos arguments:

nom és el títol que se li dóna al procediment
llista és una llista formada per subllistes: la primera subllista inclou tots el paràmetres d'entrada del procediment, les altres les instruccions necessàries per definir el procediment.

<u>Exemples</u>:
```
?defineix "poli [[costat angle] [avança :costat
gira.dreta :angle] [poli :costat :angle]]
equival a la definició usual
procediment poli :costat :angle
avança :costat gira.dreta :angle
poli :costat :angle
fi

?defineix "salutació [[][escriu "hola]]
equival a la definició usual
procediment salutació
escriu "hola
fi
```

llista.proc nom Actua de manera inversa a l'anterior. Retorna la definició del procediment nom sota el format propi de **defineix**.

<u>Exemples</u>:
```
procediment mitjana :x :y
retorna (:x + :y)/2
fi
llista.proc "mitjana  retorna
[:x :y] [retorna (:x + :y)/2]
```

copia.definició nom.vell nom.nou Copia la definició del procediment nom.vell dins del procediment nom.nou. Es tindran dos noms amb la mateixa definició. En principi, nom.vell i nom.nou no han de ser primitives.

2 ESTRUCTURES DEL LLENGUATGE

2.1 ELEMENTS DEL LLENGUATGE

2.1.1 ELEMENTS BÀSICS

Tortuga

És el nom que es dóna al petit cursor triangular que apareix al centre de la pantalla quan el LOGO està en modalitat gràfica. Els dibuixos es fan indicant a la Tortuga que es mogui sobre la pantalla deixant rastre del seu recorregut. Les instruccions bàsiques que es donen a la Tortuga són: moure's cap endavant o cap enrera en línia recta (recorrent la distància especificada) i girar a dreta o esquerra (l'angle de gir que es determini).

La posició inicial de la Tortuga és apuntant cap amunt i situada al centre físic de la pantalla.

Paraula

Una paraula és una sèrie de caràcters sense espais en blanc entremig. S'escriuen precedides per cometes (que no s'han pas de tancar).

Exemples:
"solució
"primer.valor
1992
"   (paraula buida)

Número

Els números es consideren paraules. Quan s'escriu un número no s'ha de col.locar les cometes al davant.

Llista

Una llista és una relació d'elements que poden ser paraules, números o fins i tot llistes. S'han d'escriure entre claudàtors.

Exemples:
[això és un llista]
[[gos perro] [finestra ventana]]
[] (llista buida)
[exemple [de llistes [una dins de l'altre]]]

2.1.2 VARIABLES

Una variable és una zona reservada de memòria on s'hi poden guardar, indistintament, paraules, llistes, valors numèrics,... Les variables permeten dues accions bàsiques: emmagatzemar dins seu informació i recuperar-la posteriorment.

Cada variable queda identificada per un nom que permet referir-s'hi. Els noms de les variables són paraules. Quan es fa referència a una variable el nom ha d'anar precedit per cometes: "nom, "número, "dia.mes,... Quan es fa referència al que hi ha emmagatzemat dins la variable, o sia al seu contingut, el nom ha d'anar precedit per dos punts: :nom, :número, :dia.mes,... L'exemple que es dóna tot seguit, on se suposa que la variable costat conté el valor 50, serveix per a il.lustrar la diferència entre el nom i el contingut d'una variable.

Exemples:
?escriu :costat escriurà a la pantalla 50
?escriu "costat escriurà a la pantalla costat

Per a guardar dades dins una variable s'utilitza l'ordre **posa.a**.

Exemples:
?posa.a "n 5 assigna el valor 5 a la variable n
?posa.a "p "ona assigna la paraula ona a la variable p
?posa.a "p sense.primer "ona li assigna la paraula na
?posa.a "n :n-1 assigna a la variable n un valor menor en una unitat al que tenia abans. Si aquest era 5, després d'aquesta instrucció serà 4.

Les variables d'entrada d'un procediment són variables locals. Les creades amb un **posa.a** tenen un caràcter global. Aquestes últimes al quedar dins de l'espai de treball poden crear interferències. Per fer que una variable, que en principi seria global, es transformi en local es disposa de la primitiva **fes.local**.

2.1.3 OPERADORS ARITMÈTICS I LES SEVES EXPRESSIONS

Els operadors aritmètics que incorpora el LOGO són:

Suma que s'indica amb el símbol +
Resta que s'indica amb el símbol -
Multiplicació que s'indica amb el símbol *
Divisió que s'indica amb el símbol /

Les expressions aritmètiques es formen combinant constants numèriques, operadors aritmètics, variables i funcions numèriques. El producte i la divisió tenen prioritat respecte a la suma i a la resta. Els parèntesis permeten alterar aquestes prioritats, ja que les primeres operacions que s'efectuen són les indicades entre parèntesis.

Exemples:

```
(:x - 4) / :y
arrel :x / (:y + :x)
4 + quocient 23 4
residu 23 4 + :x
```

2.1.4 OPERADORS LÒGICS I LES SEVES EXPRESSIONS

Es disposa dels operadors de comparació:

```
=   per a indicar la igualtat
<   per a indicar és menor
>   per a indicar és major
```

Els operadors lògics són: **veres.totes**, **vera.alguna** i **no**, que permeten efectuar les operacions lògiques tradicionals. Actuen sobre predicats o sia expressions que, un cop avaluades, retornen tan sols els valors **ver** o **fals**. Amb una certa freqüència aquestes expressions s'escriuen entre parèntesis per a facilitar la seva lectura.

Exemples:

```
veres.totes (:x=1) (:y<0)
(veres.totes (:x=1) (:y>0) (:z=0))
(vera.alguna (:des="ar) (:des="er) (:des="ir))
no (:a=0)
no (vera.alguna (:res="si) (:res="no))
```

2.2 REPETICIÓ

L'ordre **repeteix** permet incorporar al LOGO l'estructura repetitiva. La sintaxi d'aquesta ordre s'ha descrit a l'apartat 1.5.2.

El nombre de vegades que es repeteix l'acció pot ser constant o variable. També es poden encadenar diverses estructures repetitives.

Exemples:

```
?repeteix 4 [avança 50 gira.dreta 90]
?repeteix 4 [repeteix 4 [avança 50 gira.dreta 90]
gira.dreta 90]

procediment salutacions :número
repeteix :número [escriu "hola]
fi
```

A més a més de variar el nombre de repeticions, es pot modificar l'acció o accions. Les modificacions abans d'una nova repetició s'aconsegueixen combinant les ordres **repeteix** i **posa.a**.

<u>Exemples</u>:

```
?posa.a "v 1992
?repeteix 4 [escriu :v posa.a "v sense.primer :v]
```

```
procediment escala.decreixent :graó
repeteix 4 [avança :graó gira.dreta 90 avança :graó
gira.esquerra 90 posa.a "graó :graó - 5]
fi
```

El nombre de graons és sempre 4; cada graó nou és més petit ja que abans de cada repetició es redueix en 5 unitats la seva grandària.

El procediment següent escriurà verticalment una llista de cinc notes:

```
procediment notes.verticals
posa.a "notes [do re mi fa sol]
repeteix 5 [escriu primer :notes posa.a "notes
sense.primer :notes]
fi
```

2.3 PROCEDIMENTS. SINTAXI.

<u>Procediments</u>

Una de les característiques més destacables del llenguatje LOGO és que ofereix la possibilitat d'enriquir-lo amb noves paraules, definides per l'usuari. Un cop definida una nova paraula, aquesta es transforma en part integrant del vocabulari de treball. La definició es fa amb les ordres bàsiques que disposa el llenguatge o d'altres paraules que s'hagin definit. Aquestes definicions reben el nom de "procediments".

El nom que serveix per a identificar cada procediment es pot escriure en qualsevol idioma. S'aconsella que no sigui molt llarg i que tingui relació amb el que s'ha definit. En cas d'estar format per dues o més paraules aquestes s'han d'unir amb punts (o algun altre símbol).

<u>Exemples</u>:

```
procediment rectangle
repeteix 2 [avança 70 gira.dreta 90 avança 50
gira.dreta 90]
fi
```

```
procediment copiador
```

```
repeteix 5 [escriu [exemple de procediment]]
fi
```

Un cop definida, la paraula **rectangle** queda associada al conjunt d'ordres incloses dins la seva definició i cada vegada que s'escrigui la paraula **rectangle** o aparegui dins d'un altre procediment es dibuixarà sobre la pantalla aquesta figura.

Exemple:

```
procediment creu
repeteix 4 [rectangle gira.dreta 90]
fi
```

Sintaxi d'un procediment La definició d'un procediment consta de tres parts:

a) Títol procediment nom.del.procediment

b) Cos o seqüència d'ordres ....

c) Tancament fi

La paraula **procediment** es pot abreujar com **proc.**

2.4 PROCEDIMENTS AMB VARIABLES

En l'exemple de l'apartat 2.3 es definia un rectangle de dimensions fixes, l'altura era de 70 unitats i la base de 50 unitats. Normalment, és més útil definir procediments tals que les dimensions es puguin variar cada vegada que els fem servir. Per això s'utilitzen les variables:

Exemple:

```
procediment rectangle.var :base :altura
repeteix 2 [avança :altura gira.dreta 90 avança :base
gira.dreta 90]
fi
```

Una vegada definit el procediment **rectangle.var** si es vol dibuixar un rectangle de base 60 i d'altura 30 es donarà l'ordre **rectangle.var 60 30.**

Com a nom de variable es pot utilitzar qualsevol paraula. Cal escriure-la precedida de **:**. Es recomana que el nom no sigui gaire llarg i que recordi el que representa l'esmentada variable.

Quan dins d'un procediment hi ha més d'una variable, l'ordre d'entrada de les dades ha de coincidir amb l'ordre en què figuren les variables en la definició del procediment.

Un procediment pot incloure en la seva definició el nom de qualsevol altre procediment ja definit, i fins el nom del procediment que s'està definint (com ja es veurà més endavant). En la pràctica, és freqüent que la variable del procediment sigui utilitzada també a l'hora de cridar el procediment que l'inclou:

Exemple:

```
procediment dos.quadrats :costat
quadrat :costat
quadrat :costat / 2
fi
```

Aquest procediment utilitza un altre procediment anomenat **quadrat** que se suposa definit. La variable de **dos.quadrats** s'utilitza també en **quadrat**.

Exemples:

```
procediment copiador1 :noms
repeteix 4 [escriu :noms]
fi

procediment copiador2 :noms :vegades
repeteix :vegades [escriu :noms]
fi
```

Un cop definit el procediment **copiador2** si es vol escriure 7 vegades la paraula dijous es donarà l'ordre **copiador2 "dijous 7.**

Exemple:

```
procediment dos.copies :nom
copiador1 :nom
copiador1 sense.primer sense.últim :nom
fi
```

Aquest procediment n'utilitza un altre d'anomenat **copiador1** que se suposa definit. La variable de **dos.copies** s'utilitza també a **copiador1**.

2.5 MODULARITAT. PROCEDIMENTS TRANSPARENTS

El LOGO pel fet de ser un llenguatge amb procediments permet d'ampliar el vocabulari de treball d'acord amb les necessitats del programador. Aquesta característica el fa també molt adequat per a la programació modular.

Cada problema es redueix a problemes més senzills, i a la vegada, si és convenient, aquests es redueixen a d'altres de més simples. Així s'aconsegueix un programa modulat. La construcció d'un gran bloc a partir de blocs més petit i elementals rep el nom de construcció modular.

Exemples:

```
procediment marina
mar
barca
sol
fi
```

```
procediment joc
pregunta
comparació
resposta
fi
```

El procediment **marina** es dibuixarà a partir dels procediments **barca**, **sol** i **mar** definits prèviament. El procediment **joc** a partir de **pregunta**, **comparació** i **resposta**.

La programació modular permet de tornar a utilitzar mòduls creats anteriorment:

Exemple:

```
procediment port
sol mar barco barca far
fi
```

En cas de treballar amb procediments gràfics és convenient definir els procediments elementals de manera que siguin "transparents". Es diu que un procediment gràfic és transparent si després de la seva execució deixa la Tortuga en la mateixa posició i amb la mateixa orientació que tenia en començar-lo. Els procediments transparents faciliten molt la programació modular.

2.6 ESTRUCTURES CONDICIONALS. VARIABLE COMPROVA

El LOGO permet l'ús de l'estructura condicional senzilla i la condicional amb doble alternativa:

Si amb una alternativa

si expressió.lògica [acció]

Si en el moment d'avaluar l'expressió lògica obtenim com a resultat ver s'executa l'acció descrita dins dels claudàtors i, eventualment, es continua després el procediment. Si dóna com a resultat fals es continua directament l'execució del procediment.

Exemple:

```
procediment marina1 :dia
mar
si :dia = "assolellat [sol]
barca
fi
```

Si amb doble alternativa

si expressió.lògica [acció1][acció2]

Si a l'hora d'avaluar l'expressió lògica obtenim com a resultat **ver** s'executa l'acció descrita dins dels primers claudàtors i en cas contrari la que correspon als segons. En acabar-les es continua, eventualment, amb l'execució del procediment. S'ha de teclejar tota la línia seguida sense polsar ←┘ fins acabar d'introduir [acció2].

Exemple:

```
procediment marina2 :dia
mar
si :dia = "assolellat [sol] [núvols]
barca
fi
```

El LOGO també permet formular els condicionals amb la variable **comprova** d'acord amb l'esquema següent:

comprova expressió.lògica

Guarda el resultat de l'expressió lògica (**ver** o **fals**).

si.ver [acció1]

Si el resultat de l'últim **comprova** és **ver** s'executen les accions descrites dins els claudàtors.

si.fals [acció2]

Si el resultat de l'últim **comprova** és **fals** s'executen les accions descrites dins els claudàtors.

Exemple:

```
procediment marina3 :dia
barca
comprova :dia = "solejat
si.ver [sol]
si.fals [núvols]
mar
fi
```

2.7 PROCEDIMENTS RECURSIUS DE CUA

Són aquells procediments en els quals el nom del procediment que s'està definint forma part també del cos de la definició, ja sigui en la penúltima línia (procediments recursius de cua o simples) o en una altra posició dins del procediment (procediments recursius interns).

Els procediments següents són recursius de cua:

Exemples:

```
procediment salt
espera 20 llapis
avança 10 no.llapis
salt
fi
```

```
procediment sense.fi
escriu "hola
sense.fi
fi
```

S'observa que el resultat del primer procediment és una "Tortuga que salta" i el del segon una llista sense fi de la paraula hola que va desfilant per la pantalla. Els procediments anteriors s'han d'interrompre polsant **F10**.

També es poden definir procediments recursius de cua que tinguin variables.

Exemple:

```
procediment escala1 :graó
avança :graó  gira.dreta 90  avança :graó gira.esquerra
90
escala1 :graó
fi
```

No s'ha d'oblidar escriure el nom de la variable quan es torna a cridar el procediment. Els procediments recursius permeten alterar el valor de la variable o variables quan és fa la crida a si mateix:

Exemple:

```
procediment eko :nom
escriu :nom
eko paraula :nom últim :nom
fi
```

De moment, tots els exemples de procediments que s'han vist s'aturen amb **F10**.

2.8 CONTROL DE PROCEDIMENTS RECURSIUS

Els procediments recursius es poden aturar des de dins amb l'ordre condicional **si**, utilitzada juntament amb l'ordre **acaba**.

Les condicions o expressions lògiques, utilitzades per aturar els procediments s'han de formular de manera que necessàriament s'arribin a complir al llarg de l'execució del procediment. En cas contrari el procés no s'aturaria.

Exemples:

```
si :costat > 100 [acaba]
si :llista = [] [escriu "fi acaba]
```

```
procediment escala.decreixent2 :graó
si :graó < 5 [acaba]
avança :graó  gira.dreta 90  avança :graó gira.esquerra
90
escala.decreixent2 :graó - 5
fi
```

En aquest procediment la variable graó disminueix 5 unitats cada execució, i el procediment s'aturarà quan el valor de graó sigui inferior a 5.

Si la condició fos **si :graó = 0 [acaba]**, i el primer valor de graó fos 11, disminuint de 5 en 5, els valors que prendria serien 11, 7, 2, -3,..., però mai 0 i el procediment no s'aturaria.

Aquest tipus de control també és adequat per a procediments no gràfics:

Exemple:

```
procediment vertical :nom
si :nom = " [acaba]
escriu primer :nom
vertical sense.primer :nom
fi
```

Quan es vol comptar el nombre de vegades que es repetirà el procés, resulta convenient utilitzar una variable destinada a portar aquest control. Si se li dóna el nom "nivell", cada vegada que es torni a cridar al procediment es farà amb un valor de "nivell" disminuït en una unitat. El procés s'aturarà quan "nivell" prengui un valor més petit que 1.

Exemple:

```
procediment escala2 :graó :nivell
```

```
si :nivell < 1 [acaba]
avança :graó  gira.dreta 90  avança :graó gira.esquerra
90
escala2 :graó :nivell-1
fi
```

2.9 REPETICIÓ I RECURSIVITAT

La recursivitat de cua analitzada anteriorment, genera processos repetitius. Tots aquests es poden resoldre per mitjà de la recursivitat, si bé l'afirmació contrària no és certa: hi ha problemes que es poden resoldre utilitzant una estructura recursiva però no són adequats per a una estructura repetitiva.

El procediment **escala2** es podria torna a escriure com:

```
procediment escala3 :graó :nivell
repeteix  :nivell [avança  :graó  gira.dreta  90 avança
:graó gira.esquerra 90]
fi
```

El resultat en els dos casos és el mateix. En aquest cas la solució repetitiva és una mica més senzilla i a més a més utilitzarà, com és freqüent, menys memòria durant l'execució que la solució recursiva. Per una altra banda, l'ús de solucions recursives obre les portes a la resolució de problemes més complexos. Val la pena, doncs, de conèixer les dues mecàniques de resolució.

2.10 PROCEDIMENTS RECURSIUS (AMB UNA CRIDA INTERNA)

En els procediments recursius, la situació en què està la crida que fa a si mateix dins del cos del procediment, influeix de manera notable en el seu funcionament. Quan la crida es realitza com l'última acció del procediment es té un procediment recursiu de cua (2.7) i correspon a una repetició (2.9). Quan la crida es realitza en una posició diferent, i queden tasques per fer després de la crida recursiva, es parla de procediments recursius "amb una crida interna". Aquests són l'objectiu d'aquest apartat i per estudiar-los es recomana comparar el dos exemples següents:

Exemples:

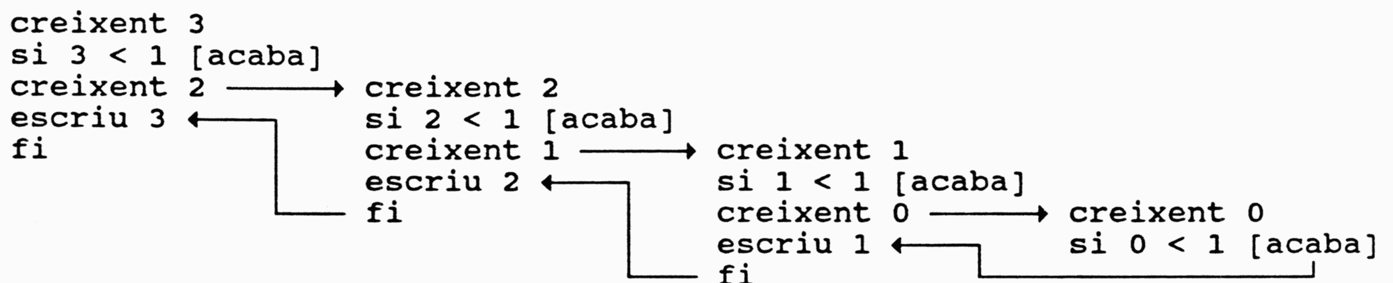
```
procediment decreixent :n
si :n < 1 [acaba]
escriu :n
```

```
procediment creixent :n
si :n < 1 [acaba]
creixent :n - 1
```

```
decreixent :n - 1
fi
```

```
escriu :n
fi
```

El primer procediment escriu, en vertical, els números 3, 2, 1 (si n = 3). El segon escriu, també en vertical, 1, 2, 3 (si n = 3). En aquest segon cas, com que la crida recursiva està dins del procediment, queden tasques pendents i els números s'escriuen en ordre invers. Aquest esquema es desenrotlla a continuació:



Si bé s'utilitzen molt les estructures recursives en la manipulació de llistes resulta més fàcil la seva comprensió amb exemples gràfics.

Ara es tracta d'escriure un procediment recursiu que dibuixi una sèrie de triangles equilàters apilats. Un dels procediments que dibuixa un triangle equilàter és:

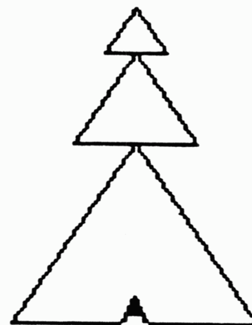
```
procediment triangle :costat
gira.esquerra 90 avança :costat/2 gira.dreta 120 avança
:costat
gira.dreta 120  avança  :costat  gira.dreta  120 avança
:costat/2
gira.dreta 90
fi
```

Les dues primeres línies situen la Tortuga en posició i dibuixen fins al vèrtex superior del triangle. La tercera i la quarta són per a completar el triangle. L'última és per a fer transparent el procediment.

Per a obtenir un arbre de triangles interessa, que en arribar al vèrtex superior, es dibuixi un arbre de triangles més petit. Per a això s'introduirà la instrucció 'arbre de triangles' després de posicionar adequadament la Tortuga.

```
procediment arbre.triangles :costat
si :costat < 10 [acaba]
gira.esquerra  90  avança costat/2 gira.dreta 90 avança
:costat
```

```
gira.esquerra 30  arbre.triangles :costat/2  gira.dreta
30
gira.dreta 120  avança  :costat  gira.dreta  120 avança
:costat/2
gira.dreta 90
fi
```



artri 80 3

Cal observar detingudament com es dibuixa aquest arbre sobre la pantalla. Primerament tot el costat esquerre i després el costat dret (tasques pendents). La línia condicional s'introdueix per a detenir la recursivitat.

Tot seguit es dóna un procediment que fa el mateix però que s'ha definit utilizant la variable nivell:

```
procediment artri :costat :nivell
si :nivell < 1 [acaba]
gira.esquerra 90  avança :costat/2 gira.dreta 120 avança
:costat
gira.esquerra 30 artri :costat/2 :nivell-1 gira.dreta 30
gira.dreta  120  avança  :costat  gira.dreta  120 avança
:costat/2
gira.dreta 90
fi
```

2.11 RECURSIVITAT GENERAL

En l'apartat 2.10 s'ha analitzat la recursivitat amb una crida interna. Quan un procediment recursiu es crida internament més d'una vegada, augmenta considerablement la seva complexitat. La potència d'aquesta nova eina és notable i es parla de recursivitat general. Las seves aplicacions dins la manipulació de llistes són importants.

Partint del procediment **artri** ens acostarem al concepte de recursivitat general d'una manera paulatina:

```
procediment artri :costat :nivell
si :nivell < 1 [acaba]
gira.esquerra 90 avança :costat/2

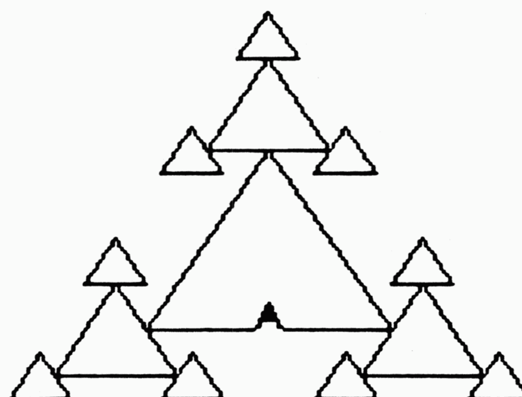
gira.dreta 120 avança :costat
gira.esquerra 30 artri :costat/2 :nivell-1 gira.dreta 30
gira.dreta 120 avança :costat

gira.dreta 120 avança :costat/2
gira.dreta 90
fi
```

Les línies en blanc corresponen als instants en què la Tortuga arriba als vèrtexs de la base del triangle. Si en aquests instants el procediment es crida a si mateix es crearà una nova estructura, de manera semblant a la que s'havia edificat sobre el vèrtex superior.

```
procediment artri :costat :nivell
si :nivell < 1 [acaba]
gira.esquerra 90 avança :costat/2
gira.esquerra 30 artri :costat/2 :nivell-1 gira.dreta 30
gira.dreta 120 avança :costat
gira.esquerra 30 artri :costat/2 :nivell-1 gira.dreta 30
gira.dreta 120 avança :costat
gira.esquerra 30 artri :costat/2 :nivell-1 gira.dreta 30
gira.dreta 120 avança :costat/2
gira.dreta 90
fi
```

La introducció dins del procediment **triangle** de la crida recursiva sobre el vèrtex superior va generar còpies decreixents de la figura sobre ella mateixa (primera versió del procediment **artri**). En el cas present, en el qual es fan tres crides recursives, es dibuixen sobre els tres vèrtexs figures arborescents anàlogues a la figura principal. La variable nivell permet regular el grau d'arborescència de la figura.



artri 80 3

2.12 CONSTRUCCIÓ DE FUNCIONS

El LOGO disposa com a primitives d'un gran nombre de funcions. Aquest nombre es pot incrementar construint-ne de noves. La construcció de noves eines suposa moltes vegades la creació de nous procediments que en moltes ocasions seran noves funcions.

Una funció pot tenir un o diversos arguments (i fins i tot cap) però pot retornar tan sols un únic objecte. Aquests objecte pot ser tan complex com es vulgui (exemple: una llista de llistes).

La primitiva **retorna** permet la creació de funcions, interrompent l'execució del procediment i retornant un valor. Quan s'utilitzen cal preveure el que es farà amb el resultat (passar-lo a un escriu, utilitzar-se dins d'un altre porcediment,...)

Exemples:

```
procediment dobles :nom
retorna frase :nom :nom
fi

procediment quadrat :x
retorna :x * :x
fi

procediment  hipotenusa :base :altura
retorna arrel (quadrat :base + quadrat :altura)
fi
```

Utilitzant algunes primitives definides anteriorment i **retorna** es pot definir el procediment **un.al.atzar** que extreu a l'atzar un element d'una llista o paraula donada:

```
procediment un.al.atzar :objecte
retorna element (1 + atzar (n.elements :objecte))
:objecte
```

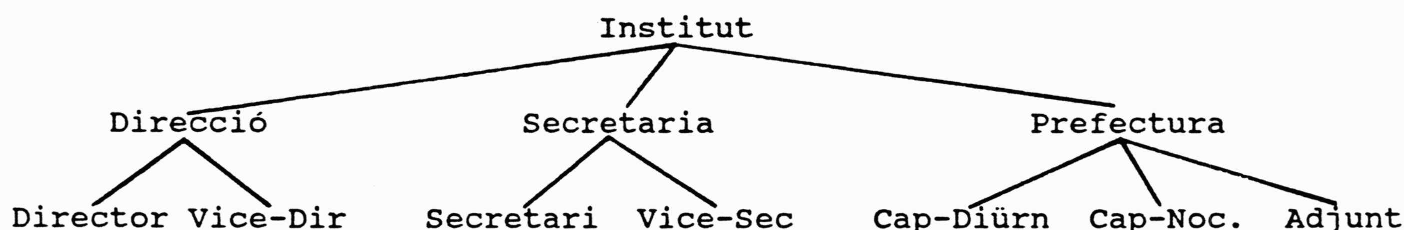
fi

Exemples:
un.al.atzar "pera pot retornar p, e, r o a
un.al.atzar [do re mi] pot retornar do, re o mi

2.13 DIVERSOS TIPUS DE LLISTES

El fet que les llistes del LOGO puguin tenir com a elements no tan sols paraules sinó també altres llistes permet manipular amb facilitat diferents tipus d'estructures.

Estructures jerarquitzades:
Per exemple l'arbre següent, que correspon als càrrecs d'un institut:



Pot representar-se per mitjà de la llista:

```
[ [Direcció     [Director  Vice-Dir]]
  [Secretaria   [Secretari  Vice-Sec]]
  [Prefectura   [Cap-Diürn  Cap-Noc.  Adjunt]] ].
```

Les primitives d'extracció (1.3.1) combinades segons les necessitats del cas permeten accedir a qualsevol del seus elements i les de composició (1.3.2) permeten la formació, de noves llistes respectant la jerarquia.

Per facilitar-ne la manipulació es poden fer les assignacions següents:

```
posa.a "dir [Direcció     [Director    Vice-Dir]]
posa.a "sec [Secretaria   [Secretari Vice-Sec]]
posa.a "pre [Prefectura   [Cap-Diürn  Cap-Noc. Adjunt]]
posa.a "institut (llista :dir :sec :pre)
```

La variable institut conté ara tota la llista anterior. S'ha resumit tota la informació de l'arbre en una sola llista.

Llistes enllaçades:

Una llista pot estar formada també per elements que són noms de variables. Aquestes llistes s'anomenen enllaçades. Per a la seva manipulació s'utilitzen, entre altres, algunes de les primitives del apartat 1.3.6.

```
posa.a  "identitat [noms cognoms]
posa.a  "noms [Pere Pau Pep]
posa.a  "cognoms [Puig Llop Tort]
```

És com si s'hagués creat la llista identitat:

[[Pere Pau Pep] [Puig Llop Tort]]

<u>Llistes d'associació</u>:

Són llistes integrades per sub-llistes, que estan formades per un primer element que representa una característica i la resta d'elements que corresponen a valors associats a ella.

<u>Exemple</u>:
[[color blau verd groc] [material fusta plàstic guix fang] [forma rodona quadrada]]

Aquest tipus de llistes també permeten la representació d'arbres.

<u>Llistes de propietats</u>:

És un cas particular de l'anterior, en el qual cada característica té associat un únic valor. Permeten associar propietats a un objecte determinat.

<u>Exemple</u>:
[[color blau] [material plàstic] [forma rodona]]

El LOGO disposa de primitives especials per al tractament de llistes de propietats (1.6.1)

3 AMBIENT LOGO

3.1 ENTRADA I SORTIDA DE L'AMBIENT LOGO

Si es disposa d'un disc de treball que permeti l'entrada directa a LOGO (en aquest cas ACTI-LOGO, versió catalana per un compatible PC), els passos necessaris per a començar a treballar són:

- Inserir el disc de treball en la unitat de discos A i tancar-la bé.
- Engegar l'ordinador.
- Apareixerà en pantalla el missatge: Acti-Logo. Quin espai de memòria vols reservar?
- Polsar ↵
- A continuació apareixerà un altre missatge: Benvingut a Logo
- S'està en condicions de començar a treballar.

En finalitzar cada sessió de treball i després d'haver enregistrat el contingut de la memòria, se surt del ambient LOGO, retornant al sistema operatiu, utilitzant la primitiva **.adéu**.

3.2 TECLAT

3.2.1 CARACTERÍSTIQUES

El teclat que es descriu aquí correspon al model Micral 30 (marca Honeywell - Bull), compatible amb el PC d'IBM. Està dividit en tres zones: la central amb les lletres, números i altres tecles; a la dreta hi ha el teclat numèric i d'edició i a l'esquerra les tecles de funció (**F1** a **F10**).

3.2.2 ZONA CENTRAL

↵ Correspon al RETURN, NEW-LINE o ENTER d'altres ordinadors. Cal polsar-lo després d'introduir cada instrucció o conjunt d'instruccions. No s'utilitza per al salt de línia que és automàtic als ordinadors.

Equival a la tecla de majúscules d'una màquina d'escriure convencional.

Ctrl S'utilitza combinant-la amb d'altres tecles per a produir uns certs efectes.

Alt Permet l'accés a uns certs caràcters i accions específiques, quan es combina amb d'altres tecles.

←	Tecla grisa superior. Mou el cursor cap a l'esquerra esborrant els caràcters per allà on passa.
Ctrl/Alt/[	Polsades simultàniament permeten escriure el claudàtor obert.
Ctrl/Alt/]	Polsades simultàniament permeten escriure el claudàtor tancat.

3.2.3 ZONA NUMÈRICA

Del	Esborra el caràcter sobre el qual es troba el cursor.
Num Lock	Serveix per activar i desactivar el teclat numèric. Quan està activada, amb la llum encesa, serveix per entrar els números. Quan està desactivada serveix per fer moure el cursor.
Ctrl/Num Lock	Atura momentàniament una execució. Aquesta es reprèn polsant una tecla qualsevol.
→ ←	Mouen el cursor cap a la dreta i cap a l'esquerra respectivament. (Tecles blanques)
↑ ↓	Si s'està a dins l'editor mouen el cursor amunt i avall respectivament. (Tecles blanques).
/Prtsc	Imprimeix el contingut de la pantalla.

3.2.4 TECLES DE FUNCIÓ

F1	Sortida de l'editor enregistrant el seu contingut. Passa a pantalla de text.
F2	Passa a pantalla mixta
F3	Recupera l'última línia teclejada
F4	Passa a pantalla gràfica
F5	Pausa (Veure 1.5.5)
F6	Recerca dins l'editor. Crtl/F6 serveix per a continuar
F7	Recerca i canvia: T = tots, D = un a un
F8	Editar més procediments
F9	Portar arxius a l'editor

F10 Interromp el procés que està en curs. Per a sortir de l'editor sense enregistrar el seu contingut

Consultar el manual per a més detalls sobre les tecles **F6, F7, F8 i F9.**

3.3 TIPUS DE PANTALLES

Quan es treballa amb LOGO la pantalla pot estar en tres estats diferents: sols text, gràfics més text o sols gràfics. A continuació es descriuen breument cadascun d'aquests estats i s'indica com accedir-hi.

Pantalla mixta Gràfics més text. La Tortuga apareix al seu centre geomètric. Tota la pantalla està dedicada a gràfics menys les 7 últimes línies que estan reservades a text. Aquestes línies permeten de veure les ordres teclejades i també els missatges d'error. Si el dibuix és prou gran se superposa al text. Aquestes set línies de text tenen efecte d'enrotllat, o sia que es van desplaçant automàticament cap amunt per fer lloc al text que es va introduint. Per entrar en aquesta modalitat cal teclejar qualsevol de les ordres gràfiques, **F2** o **mixt.**

Pantalla gràfica Sols gràfics. La Tortuga apareix al centre de la pantalla. Tota la pantalla està destinada a gràfics i no hi ha espai per a visualitzar les ordres que es van entrant. De tota manera, encara que no es puguin veure les ordres, aquestes s'executen de la forma habitual. Quan apareix un missatge d'error es passa, de forma automàtica, a la pantalla mixta. Per entrar en aquesta modalitat calen les ordres **F4** o **gràfic.**

Pantalla de text Sols text. Tota la pantalla està dedicada a text i no apareix la Tortuga. És l'estat inicial del sistema i el que correspon a la sortida de l'editor. Si s'entra qualsevol ordre gràfica es passa automàticament a pantalla mixta. Per entrar en aquesta modalitat calen les ordres **F1** o **text.**

3.4 EDITOR D'ORDRES

És freqüent que teclejant les ordres es produeixin errors. A vegades aquests es poden corregir abans de polsar ⟵┘ i d'altres vegades és el sistema Logo el que ens avisa de la seva existència. Tot seguit es donen algunes recomanacions d'interès:

<u>Tecleig</u>

Es poden teclejar conjuntament diverses ordres sense que calgui polsar ←┘ després de cadascuna. L'única limitació és que la longitud total del text no sobrepassi els 128 caràcters.

S'ha d'incloure almenys un espai en blanc entre ordre i ordre i entre ordre i argument.

L'última línia executada es pot recuperar íntegrament, sense que s'hagi de teclejar novament, polsant **F3**.

<u>Correcció</u>

En el cas que hi cap error aquest es pot corregir utilitzant les tecles adequades. Les tres accions bàsiques per a la correcció es descriuen a continuació.

<u>Tecles per a la correcció</u>

←—— Tecla grisa. Esborra el caràcter situat a l'esquerra del cursor. Si s'utilitza repetidament es retrocedeix al llarg del text esborrant tot el que es troba durant el recorregut.

← Tecla blanca. Es troba sobre la zona numérica. Permet retrocedir sense esborrar.

→ Tecla blanca. Per avançar cap a la dreta. No permet avançar més enllà de l'últim caràcter teclejat. Està sobre la zona numèrica.

Tab Porta el cursor al final de la línia.

/ Tab Porta el cursor al començament de la línia.

Del Esborra el caràcter situat sota el cursor.

<u>Mecanismes per a la correcció</u>

Les tecles descrites anteriorment serveixen per a fer les tres accions bàsiques per a la correcció:

<u>Esborrar</u>

Per esborrar un text s'ha de col.locar el cursor immediatament després d'aquest i esborrar caràcter a caràcter amb la tecla grisa ←——.

<u>Inserir</u>

Per inserir un text cal situar el cursor, amb les tecles ← i →, en el punt d'inserció i teclejar el text que es vol afegir.

<u>Canviar</u>

Per canviar un text per un altre cal primerament suprimir el text que no interessi i després, tot seguit, inserir-hi el nou.

3.5 EDITOR DE PROCEDIMENTS

3.5.1 TECLEIG

El tecleig d'un procediment es pot fer directament sobre la pantalla mixta. En el moment en què es tecleja la paraula **procediment** i el nom que se li dóna, el cursor ? es transforma en > per indicar que s'està en modalitat de memorització i no d'execució immediata. De tota manera, resulta més recomanable fer ús d'una pantalla especial que rep el nom d'Editor Logo, ja que ofereix més facilitats per a la correcció del que s'hi escriu. El camí a seguir per a definir un procediment dins l'editor és:

a.- Teclejar **edita** "nom.procediment per entrar a l'editor. Així es passa a la pantalla "Editor Logo". Hi trobarem escrit '**procediment** "nom.procediment ' i '**fi**'.

b.- Inserir el text de la definició. Al final de cada línia lògica polsar ↵. Si cal fer correccions es poden utilitzar el recursos descrits a l'apartat 3.4.

c.- Si es vol es poden teclejar més procediments sense sortir de l'editor, escrivint-los tot seguit.

d.- Per sortir de l'editor enregistrant el que s'ha definit, polsar **F1**; i si no es vol enregistrar, polsar **F10**. En el primer cas apareix a la pantalla el missatge '**acabes de definir ...**' tantes vegades com procediments s'hagin definit i amb els noms corresponents.

e.- Per comprovar la definició cal cridar el procediment des de fora de l'editor.

3.5.2 CORRECCIÓ

Entrada a l'editor

Per a corregir un procediment cal en primer lloc recuperar la seva definició. Per aconseguir-ho es tecleja la instrucció:

edita "nom.procediment

apareixerà l'editor amb la definició del procediment cridat. Cal col.locar les cometes davant del nom del procediment. (Atenció: no s'han de tancar les cometes).

Si es volgués la definició d'una sèrie de procediments es teclejaria

edita [llista de procediments]

En el cas de la llista els nom no necessiten cometes. Si es tecleja **edita** sense cap argument apareixerà l'editor en el mateix estat en que se'n va sortir.

Tecles de l'editor

A més a més de les tecles descrites per a ordres, hi ha algunes tecles que són pròpies de l'editor i que s'expliquen tot seguit:

Sobre la zona numèrica (Num/Lock desactivat):

Ins Insereix una línia en blanc a partir de la posició del cursor.

↑ Desplaça el cursor cap amunt.

↓ Desplaça el cursor cap avall.

PgUp Accedeix a la pàgina anterior de text.

PgDn Accedeix a la pàgina següent de text.

Ctrl → Esborra els caràcters situats a la dreta del cursor fins al final de la línia. Aquests caràcters és poden recuperar utilitazant **Ctrl ←**.

Ctrl ← Insereix la sèrie de caràcters esborrats amb **Ctrl →**, en el punt on es trobi el cursor. Es pot copiar una mateixa sèrie de caràcters tantes vegades com es vulgui.

Procés de correcció

Amb les tecles anteriors es poden realitzar les tres accions bàsiques per a la correcció (esborrar, inserir i canviar) explicades a l'apartat 3.4.

Sortida de l'editor

F1 Si es vol sortir enregistrant les correccions que s'han fet.

F10 Si no es volen enregistrar.

3.6 GESTIÓ DE LA MEMÒRIA DE TREBALL

3.6.1 VISUALITZACIÓ

títols? Visualitza sobre la pantalla la relació de noms de tots els procediments que hi ha dins la memòria de treball de l'ordinador en un moment donat.

presenta [llista procediments]
presenta "nom.procediment

Visualitza sobre pantalla les definicions dels procediments (o procediment) anomenats.

edita [llista procediments]
edita "nom.procediment
ed

Passa a la pantalla de l'editor mostrant la definició de tots els procediments de la llista (o procediment) sol.licitats.

3.6.2 ESBORRAT

oblida [llista procediments]
oblida "nom.procediment

Elimina els procediments (o procediment) de la memòria de treball, esborrant tots els seus noms (nom) de la llista de paraules definides.

oblida.grup "nom.de.grup

Esborra el grup especificat de l'espai de treball (fins i tot si el grup està protegit)

oblida.tot Esborra tota la memòria de treball, eliminant-hi tots el procediments i variables que contingués excepte els que hagin estat amagats per mitjà d'un **protegeix**. Quan es comença un treball nou és convenient "fer neteja" de la memòria abans de carregar un nou arxiu, si no es poden presentar problemes tant per falta de memòria com per interferències entre arxius.

3.6.3 CAPACITAT DE MEMÒRIA

En LOGO la memòria està organitzada en 'nodes' (cada node equival a 5 octets). Per conèixer la capacitat de memòria, així com per utilitzar-la al màxim es tenen les primitives:

mem.lliure Retorna un número que indica la quantitat de nodes disponibles a la memòria.

compacta Efectua una acció de 'compactat' de la memòria de treball a fi de poder disposar d'una major capacitat.

3.7 <u>GESTIÓ DEL DISC</u>

3.7.1 <u>DISCOS</u>

Per a totes les accions que es descriuen a continuació cal que el disc personal de treball es trobi dins la unitat A.

<u>Inspecció del contingut del disc</u>

arxius? Permet inspeccionar el contingut del disc. Per pantalla apareix la llista de tots els identificadors utilitzats per als arxius. Els noms dels arxius no poden tenir més de 8 lletres.

Els noms dels arxius enregistrats apareixen seguits d'una extensió: un punt seguit de fins a tres lletres. Les extensions les crea el LOGO de manera automàtica i no s'han d'escriure ni a l'hora s'enregistrar l'arxiu ni a l'hora de recuperar-lo. Si els arxius contenen procediments, paquets o el contingut de l'editor, l'extensió és **.log**. Si són gràfics l'extensió és **.im**.

<u>Canvi de la unitat activa</u>

disc Retorna la lletra que representa la unitat activa.

fes.disc Fixa la unitat lectora del disc (A, B, C, ...) com unitat activa.

<u>Exemple</u>:
?fes.disc "C:

<u>Esborrar un arxiu</u>

elimina "identificador

Esborra tot el contingut de l'arxiu i també el seu nom.

3.7.2 <u>PROCEDIMENTS</u>

<u>Enregistrament</u>

desa "identificador **procs**

Tots els procediments que estaven a la memòria de treball, excepte els que estan 'amagats' quedaran enregistrats al disc dins del mateix arxiu i identificats amb un nom únic.

desa "identificador **frase procs vars**

A més dels procediments, enregistra també les variables.

Recuperació

recupera "identificador

El nom de l'identificador ha de coincidir amb el de l'arxiu que es vol recuperar. Després de la càrrega, totes les definicions de l'arxiu passen del disc a la memòria de treball i poden ser usades de forma idèntica a com es feia en el moment de ser enregistrades.

3.7.3 PAQUETS

Enregistrament

desa "identificador "nom.del.grup

Guarda el contingut del grup "nom.del.grup dins l'arxiu "identificador

Recuperació

recupera "identificador "nom.del.grup

Recupera el contingut de l'arxiu "identificador i l'assigna a un grup anomenat "nom.del.grup.

3.7.4 GRÀFICS

Enregistrament

desa.imatge "identificador

Guarda una còpia de la pantalla a l'arxiu "identificador.

Recuperació

recupera.imatge "identificador

Recupera la imatge prèviament carregada dins "identificador.

3.7.5 EDITOR

Enregistrament

desa.editor "identificador

Guarda el contingut de l'editor directament a l'arxiu "identificador, sense que passi a la memòria de treball.

Recuperació

edita.arxiu "identificador

Carrega directament a l'editor el contingut de "identificador. La grandària d'aquest arxiu no pot sobrepassar el de l'editor; en cas contrari, s'obté el missatge **editor ple.** També es pot fer des de dins de l'editor pitjant **F9.**

3.8 IMPRESSIÓ

3.8.1 DIFERENTS POSSIBILITATS

La impressió des de LOGO es pot fer de dues maneres: només per impressora, que és el més habitual, o combinant l'aparició dels resultats per impressora amb l'aparició simultània per la pantalla.

3.8.2 PER IMPRESSORA

desa "lpt1 [llista procediments]
desa lpt1 "nom.procediment

Llista per impressora la definició dels procediments anomenats a la llista (o procediment).

desa "lpt1 procs

Llista per impressora la definició de tots els procediments que hi ha dins l'àrea de treball.

desa "lpt1 frase procs vars

Llista per impressora la definició de tots els procediments i el contingut de totes les variables.

desa.imatge "lpt1

Imprimeix el contingut de la pantalla, que pot ser dibuix o text. També es pot utilitzar la tecla / **PrtSc.** Per imprimir gràfics cal donar, des del sistema operatiu l'ordre GRAPHICS. La grandària del dibuix fet per la impressora és més reduïda si es treballa a la pantalla de 40 columnes.

desa.editor "lpt1

Llista per impressora el contingut de l'editor.

3.8.3 <u>PER IMPRESSORA I PANTALLA</u>

connecta "lpt1

Es tecleja aquesta expressió per a preparar la impressió simultània.

presenta "nom.procediment

Llista el procediment anomenat.

presenta procs

Llista tots els procediments.

presenta vars

Llista totes les variables.

nom.procediment (↵)

Imprimeix tot el que apareix a la pantalla quan s'executa el procediment cridat.

desconnecta

Es tecleja aquesta expressió per a desactivar la impressió simultània.

4 ÍNDEX ALFABÈTIC DE PRIMITIVES